Precis

Poster

Agenda

Model Driven Architectures and UML Performance Modeling Capability – Design and Usage

Leonard Weinberg, Harald Pschunder, and ***Michael Stebnisky***
Lockheed Martin MS&S
Phone: 856-722-2078
Email Address: Leonard.Weinberg@lmco.com

Both the commercial and the military markets are being driven by performance requirements that far exceed the capabilities of a minimum set of computing architectures. In addition, the requirements on many newer systems are being expressed in UML, a requirements modeling language that enforces standard and consistent practices for systems and software engineering design. There is a need to justify the computing architecture designs, and to estimate the computing architecture performance for these systems. Spreadsheets and analytical methods alone are insufficient because of the statistical nature of both the messaging and the computer operating systems. This paper describes a performance modeling tool that uses an event driven design to enable evaluation of the performance of computing architectures for which the requirements are expressed in UML.

There are many computer architecture performance modeling tools on the market. However, most tools are limited in one or more of three areas: 1-a tool may operate at a very low component level making it difficult for the system engineers to use it; 2-a tool may lack a user-friendly graphical front end, making it difficult for a designer to share the modeling tool design and model results with system engineers and customers not intimately familiar with the tool; and 3-a tool may lack any compatibility with UML requirements driven methodologies. The performance modeling capability described in this paper overcomes these three limitations, while providing a robust means for estimating the suitability of UML requirements being implemented in a computing architecture.

Before we began to design the performance modeling tool, we established three goals. First, we required the ability to predict the best design for a computing architecture that achieves satisfactory performance. Second, we sought compatibility with object oriented analysis and design methods, like UML, while not precluding other approaches. Third, we wanted an open front end that would make the tool directly accessible not only to modelers, but to system engineers, software designer, and even customers.

For the last eight years, our team has been evaluating the performance of computing architectures performing critical military applications. We have been using the BONeS event driven modeling tool. In our opinion this tool far exceeded others on the market because of the low component level available which allows us to emulate computer operations. BONeS has been discontinued, and we are currently using a Lockheed Martin product called CSIM. While BONeS and CSIM both provide the lower level component capability to emulate computer operations at reasonable level of detail, these tools can also be daunting in the amount of detail that must be specified.

In order to raise the level of detail in the model design and to speed models construction, we have introduced Infrastructure/Architecture Assemblies. (This addresses model limitation Point-1 above.) An Assembly represents message flows through internet protocols, middleware, and the

Report Documentation Page	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE **01 FEB 2005**	2. REPORT TYPE **N/A**	3. DATES COVERED **-**
4. TITLE AND SUBTITLE **Model Driven Architectures and UML Performance Modeling Capability Design and Usage**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Lockheed Martin Maritime Systems & Sensors, Moorestown, New Jersey**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release, distribution unlimited**		
13. SUPPLEMENTARY NOTES **See also ADM001742, HPEC-7 Volume 1, Proceedings of the Eighth Annual High Performance Embedded Computing (HPEC) Workshops, 28-30 September 2004. , The original document contains color images.**		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**UU**	**20**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

components. Assemblies are chosen and connected in tandem to represent sequential message flows in a UML sequence diagram. If there are ten messages in a sequence diagram, then the appropriate ten Assemblies are chosen and connected together. Our Assembly design is a perfect match to the messages in a sequence diagram. (This addresses model limitation Point-3 above.)

In our work, four basic Assembly types have been satisfactory in representing message flows. The four types of Assemblies represent: messages entering a computer and being processed in a component, messages being processed by a component and leaving a computer, messages beginning within a computer and entering another component in the same computer to be processed, and messages entering a computer to be processed and then exiting the computer. The "personality" of each Assembly is specified by completing about ten menu-based parameters. These parameters consist of: Infrastructure/Architecture Assembly type; scenario and message information; message acknowledgement on/off; component application processing time; component application priority; node assignment; and possibly network switch port connectivity.

We estimate the time to build a model using Infrastructure/Architecture Assemblies at approximately 15% of what was typically required for model development "from scratch". A typical Assembly consists of about 40 elementary component modeling blocks and 25 default parameters settings. The design and setting the default parameters are performed one time. Each time the Assembly is instantiated typically only 10 parameters are re-set.

Recently, we have developed an export utility that extracts requirements developed using UML-based commercial tools. The extracted UML requirements information is used to support performance modeling, and introduces a user friendly interface to the model design. (This addresses model limitation Point-2 above.) We have also written a CSIM utility that makes available selected UML sequence diagram flows and generates a partially completed spreadsheet containing the architecture details for the model. UML message requirements appear in the spreadsheet. Other message attributes are added by the system engineering and computer programmers. A completed spreadsheet can be made into a static model of the system, which estimates minimum message latencies and contention-free CPU utilizations. UML requirements which we extract for the spreadsheet are: message flows present in the UML sequence diagrams; UML activity diagrams to help in selecting the appropriate sequences; and node allocation information from the UML deployment diagrams.

A critical and special feature supported by our modeling tool CSIM is the ability to build the performance model one sequence diagram at a time, each independent of the other sequences. Contention among the messages for the limited CPU resources is managed by the scheduling rules we apply to the CPU resource model. Both real time priority driven preemptive scheduling and non-preemptive time-share scheduling have been modeled.

From the point of view of the software designer, our modeling trade studies attempt to minimize the number of computing resources, while providing for long term system growth and meeting critical message latency requirements under full scenario conditions. Models are run under realistic computer program priority assignments, and use benchmark estimates for the computing platform protocol stacks, middleware and application software. One class of exciting results generated from our simulations is process timelines. These are similar to sequence diagrams but they include the message latencies due to the CPU and network contentions.

Abstract

Agenda

Model Driven Architectures and UML Performance Modeling Capability – Design and Usage

Harald Pschunder and Leonard Weinberg
Lockheed Martin Maritime Systems & Sensors, Moorestown, New Jersey

Michael Stebnisky
Lockheed Martin Advanced Technology Laboratory, Cherry Hill, New Jersey

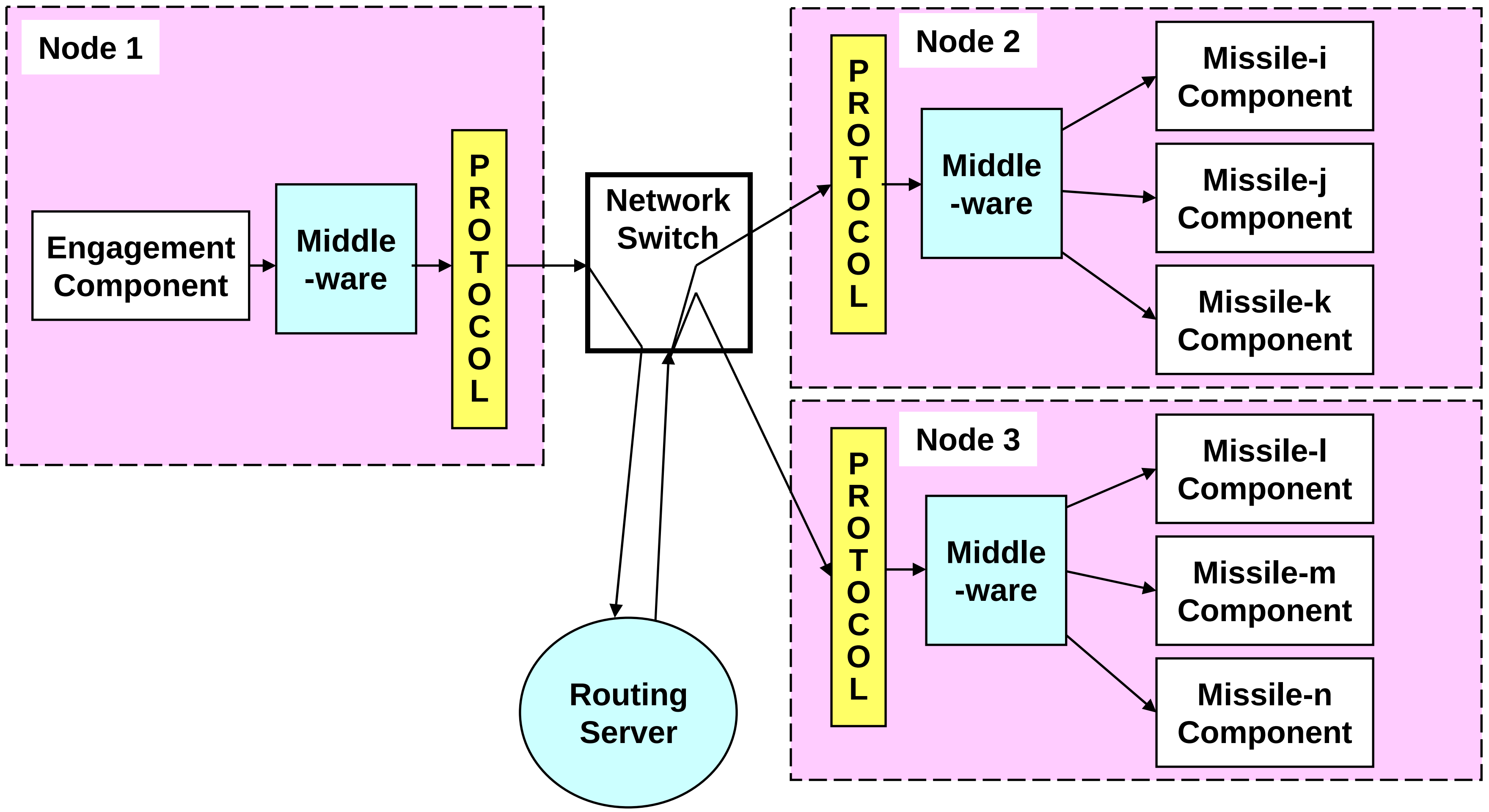
Presented at: HPEC 2004

■**September 30, 2004**

Introduction – Why is this Capability Important

- Lockheed Martin has more than 30 years experience in designing and building computing systems for U.S. Navy cruisers and destroyers
- Systems are large and demanding (12,000,000 SLOC in >50 computers)
  - Many use real-time O/S
  - Computer utilization >50 %;
  - Message latencies in the milliseconds
  - Automatic reconfiguration within seconds of failure
- Over the last eight years, event driven computing system architecture models have helped shape the computer program designs and to predict and map their performance on target systems
- For our next generation systems, we have begun development of the architectures using UML to analyze and document requirements
- For the future, we need to build a framework which makes it possible to quickly estimate and predict the dynamic performance of our future UML designed systems, and share these results with our technical community

Typical Computing Architecture Components and Communications

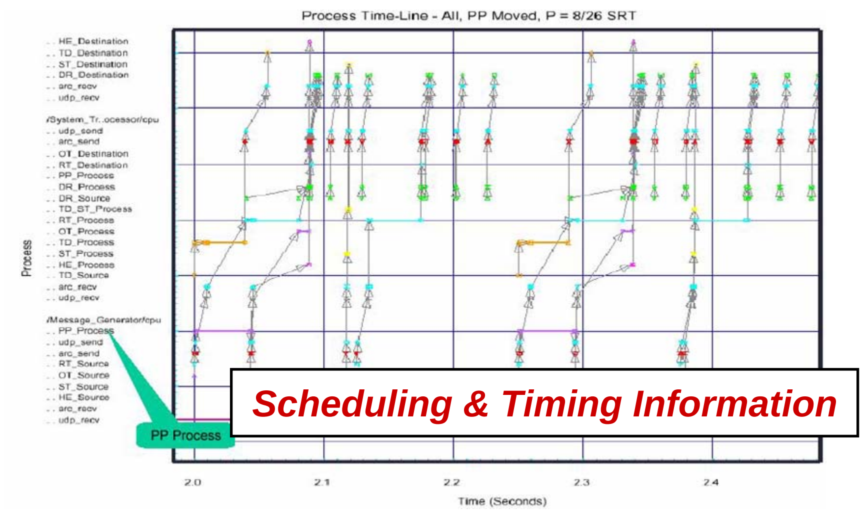


Capability of Our Performance Model

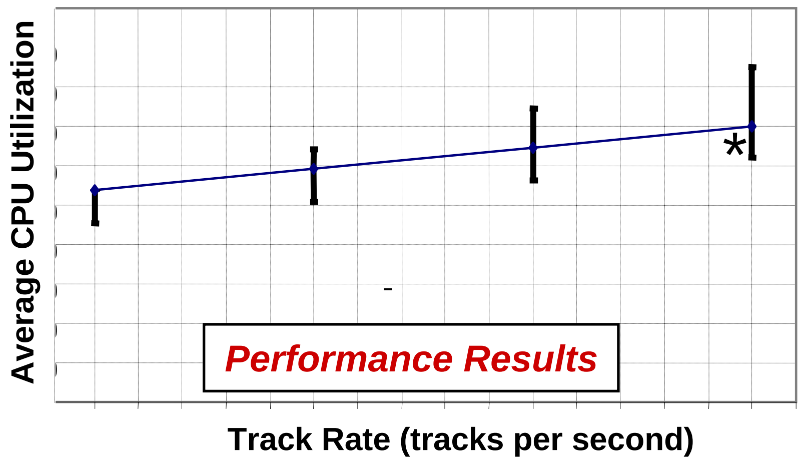
- Speeds and automates the design of performance modeling using pre-designed, off-the-shelf, large infrastructure components (modeling assemblies)
  - Eight general-purpose Infrastructure Modeling Assemblies (IMAs) were built to emulate any message's creation, flow and processing
  - The specific "personality" assumed by an IMA in a particular model is specified by completing approximately ten menu-based parameters
  - Assemblies are chosen and connected to represent any message flow
- Complies with the UML requirements modeling language
  - Our newly designed Export Conversion Program captures selected requirements and architectural information from the UML requirements models
- Incorporates a friendly front end, useable by the model designer, the system engineer and the customer
  - Sequence diagrams and spreadsheets provide the user with copies of UML requirements to build or view the performance model
  - The spreadsheet calculator also generates an estimate of model utilization and latency to help verify the performance model design

Lockheed Martin Uses the CSIM Modeling Tool

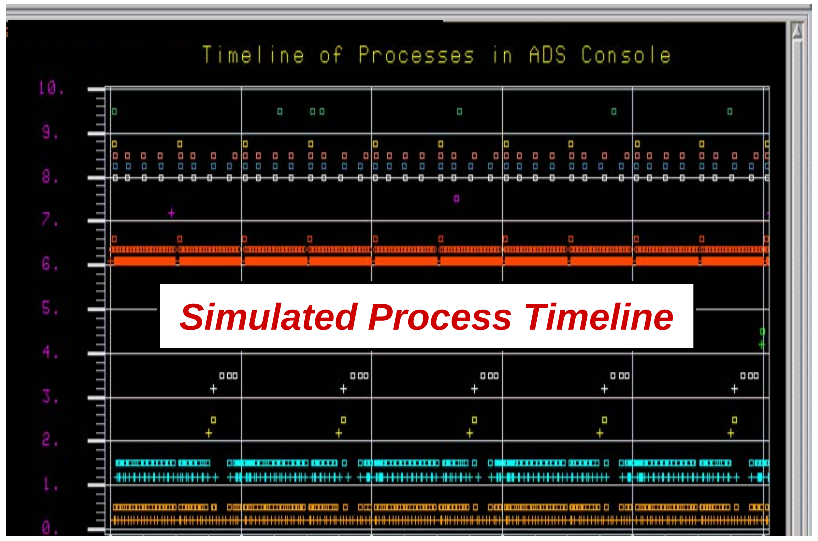
Typical Performance Modeling Results



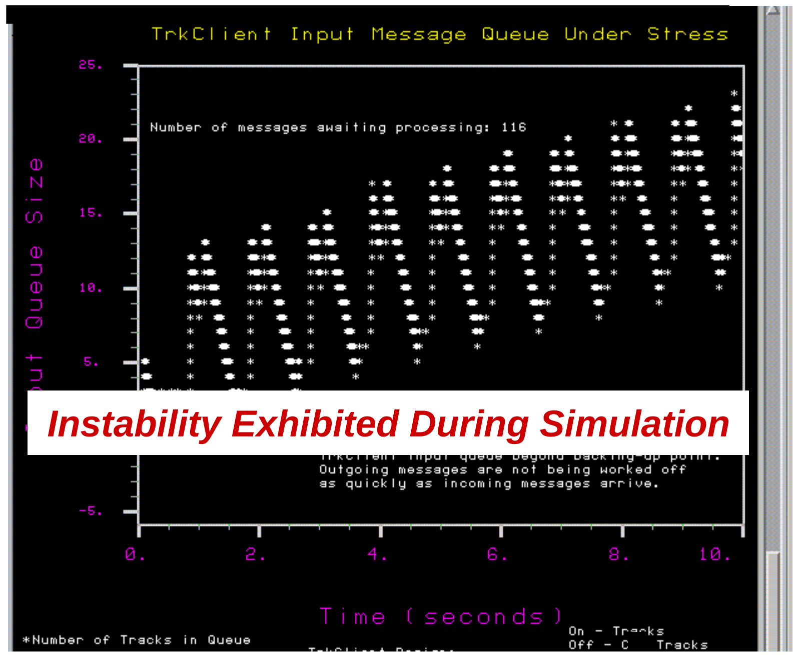
Scheduling & Timing Information



Performance Results



Simulated Process Timeline

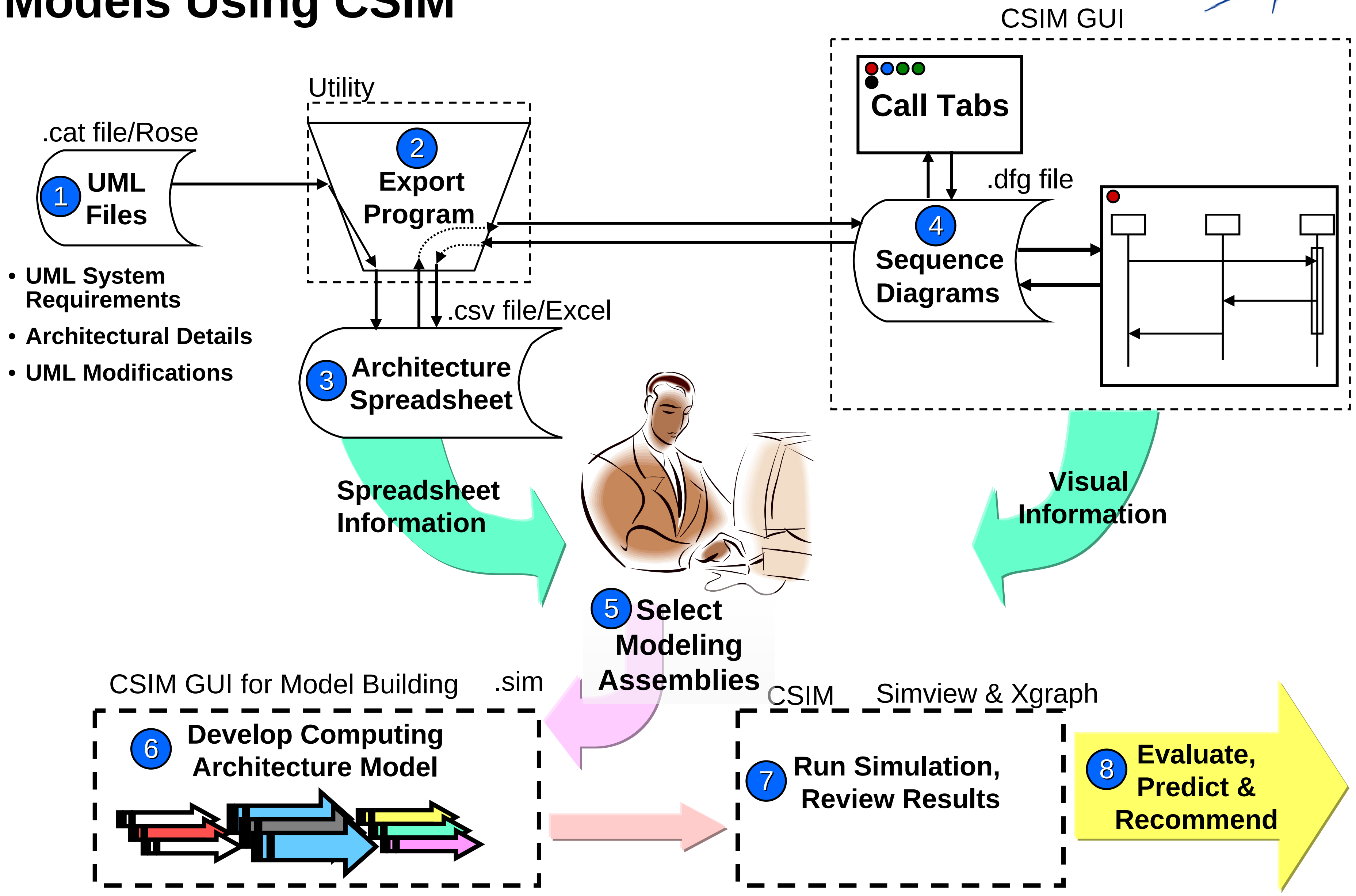


Number of messages awaiting processing: 116

TrkClient input queue beyond backing-up point.
Outgoing messages are not being worked off
as quickly as incoming messages arrive.

Instability Exhibited During Simulation

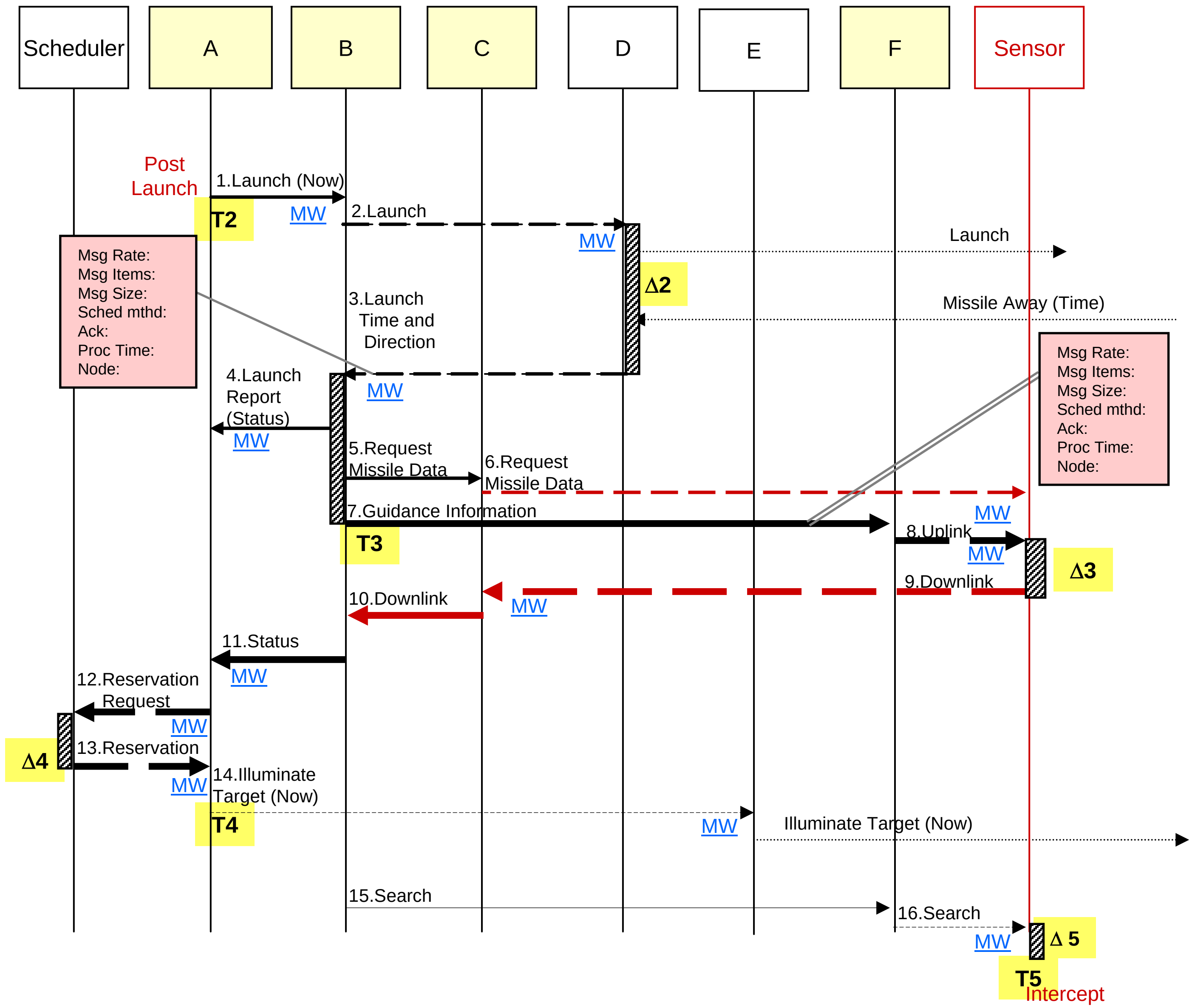
Building and Executing Performance Models Using CSIM



.cat file/Rose
1 UML Files
Utility
2 Export Program
.csv file/Excel
3 Architecture Spreadsheet
CSIM GUI
Call Tabs
.dfg file
4 Sequence Diagrams
Spreadsheet Information
Visual Information
5 Select Modeling Assemblies
CSIM GUI for Model Building
.sim
6 Develop Computing Architecture Model
CSIM
Simview & Xgraph
7 Run Simulation, Review Results
8 Evaluate, Predict & Recommend

- UML System Requirements
- Architectural Details
- UML Modifications

Example: UML Sequence Diagram with Added Architecture Detail

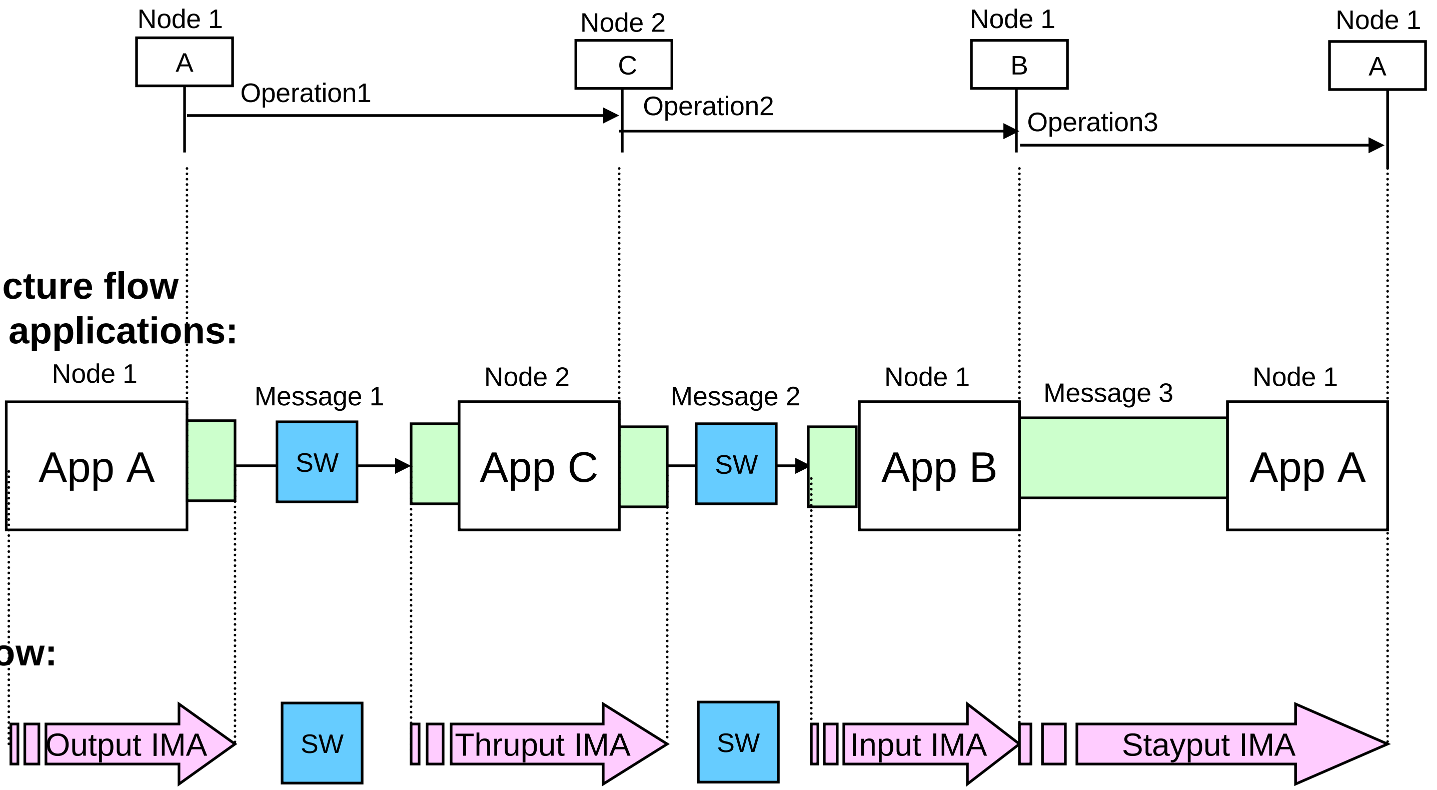


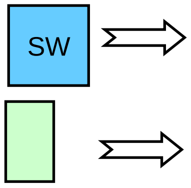
Architectural Information Used by the Performance Models

- Node Identification
- Sources and Destinations
- Message Name and Routing
- Message Size and Rate
- Message Acknowledgment
- Application Processing Time and Priority
- Software Component Scheduling Method: Real-time, Timeshare, FIFO
- etc

Sequence Diagram Flows can be Interpreted in Terms of Infrastructure Modeling Assemblies (IMAs)

- **Operations on the sequence diagram:**
- **Infrastructure flow between applications:**
- **Model flow:**





SW ⇒	Network Switch
🟩 ⇒	Middleware &/or internet protocol

An Infrastructure Modeling Assembly (IMA)

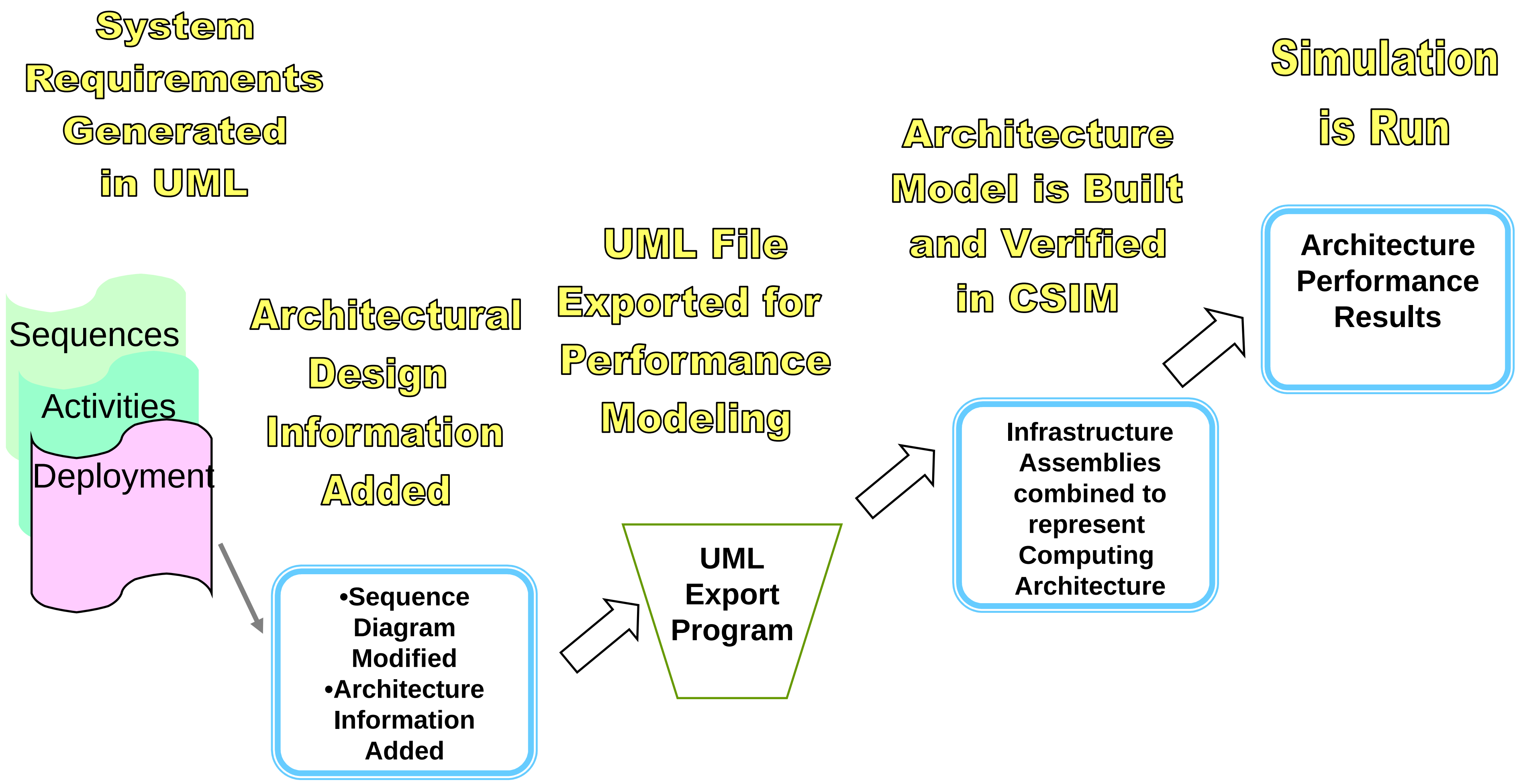
- The IMA is a model of a reasonably large infrastructure assembly, representing the processing flow initiated by the transmission of a single message
  - It may include processing by an application, middleware, and other infrastructure components and be governed by internet protocol, priority and scheduling rules
  - The IMA is built around a CPU-like resource allowing parametric control of such activities as scheduling, context switching, priority levels, managing queues, internal processing, and message input/output
- IMAs simplify building the performance model
  - We reuse these IMAs and give individual instances ‘personality’ by inserting a small number of menu-driven parameters to provide their architectural information
  - By connecting these IMAs, we emulate a Sequence Diagram of any complexity
  - Each sequence is built separately, and is independent of others until they are combined at simulation run time

We Use CMIS, a Lockheed Martin Event-Driven Simulation Tool

The Savings When Using IMAs

- **Experience indicates the large savings possible by modeling with and re-using** Infrastructure Modeling Assemblies
  - **For example, the Input** IMA **contains**
    - **~ 40 elementary blocks <u>assembled once</u>**
    - **~ 25 default parameters <u>set once when built</u>**
    - **~ 10 parameters set <u>each re-use</u>**

From UML Requirements to Computing Architecture Performance


System Requirements Generated in UML
Sequences
Activities
Deployment
Architectural Design Information Added
•Sequence Diagram Modified
•Architecture Information Added
UML File Exported for Performance Modeling
UML Export Program
Architecture Model is Built and Verified in CSIM
Infrastructure Assemblies combined to represent Computing Architecture
Simulation is Run
Architecture Performance Results

LOCKHEED MARTIN

Poster

Agenda

Model Driven Architectures and UML Performance Modeling Capability – Design and Usage

Harald Pschunder and Leonard Weinberg
Lockheed Martin Maritime Systems & Sensors, Moorestown, New Jersey

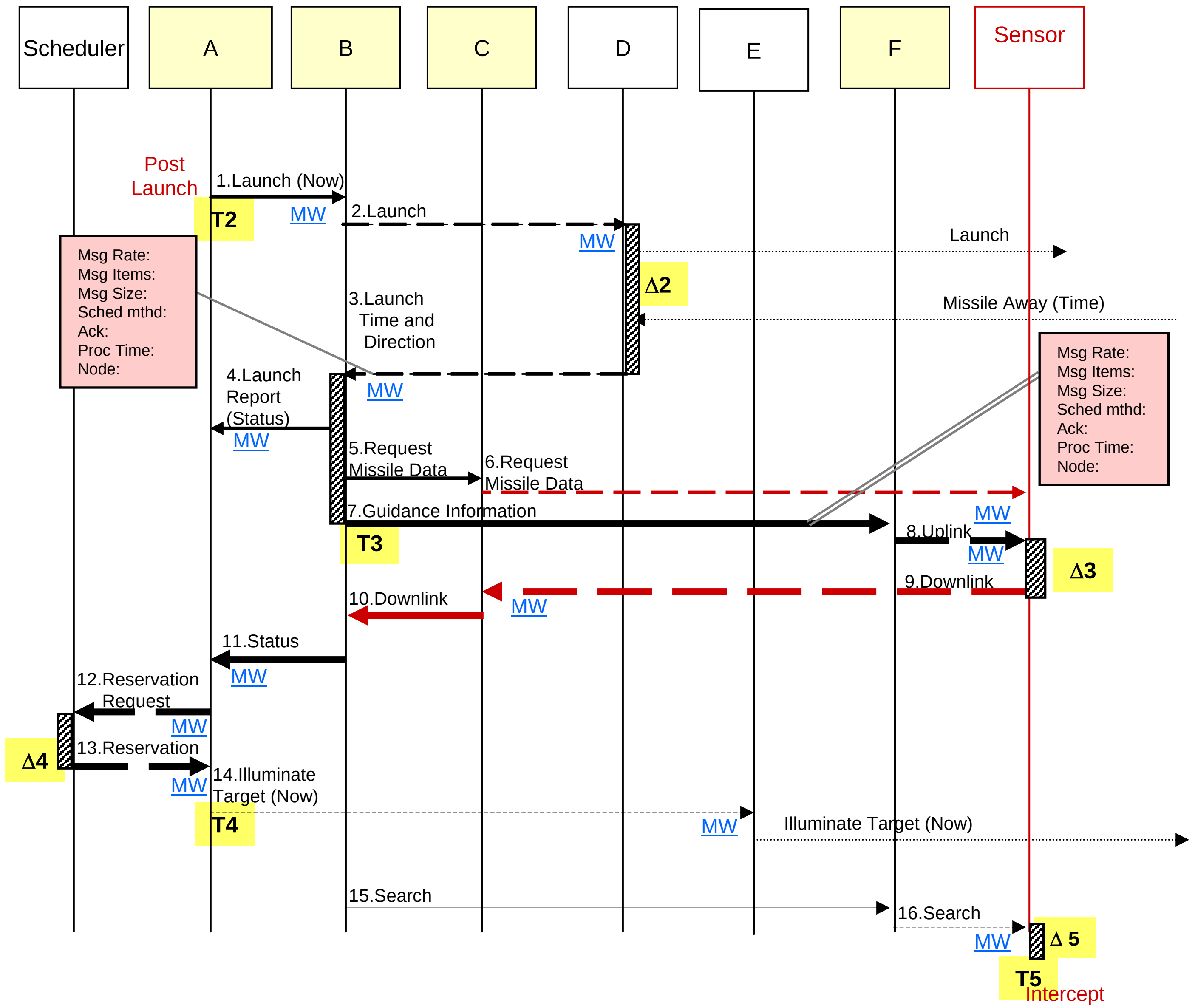
Michael Stebnisky
Lockheed Martin Advanced Technology Laboratory, Cherry Hill, New Jersey

Presented at: HPEC 2004

LOCKHEED MARTIN

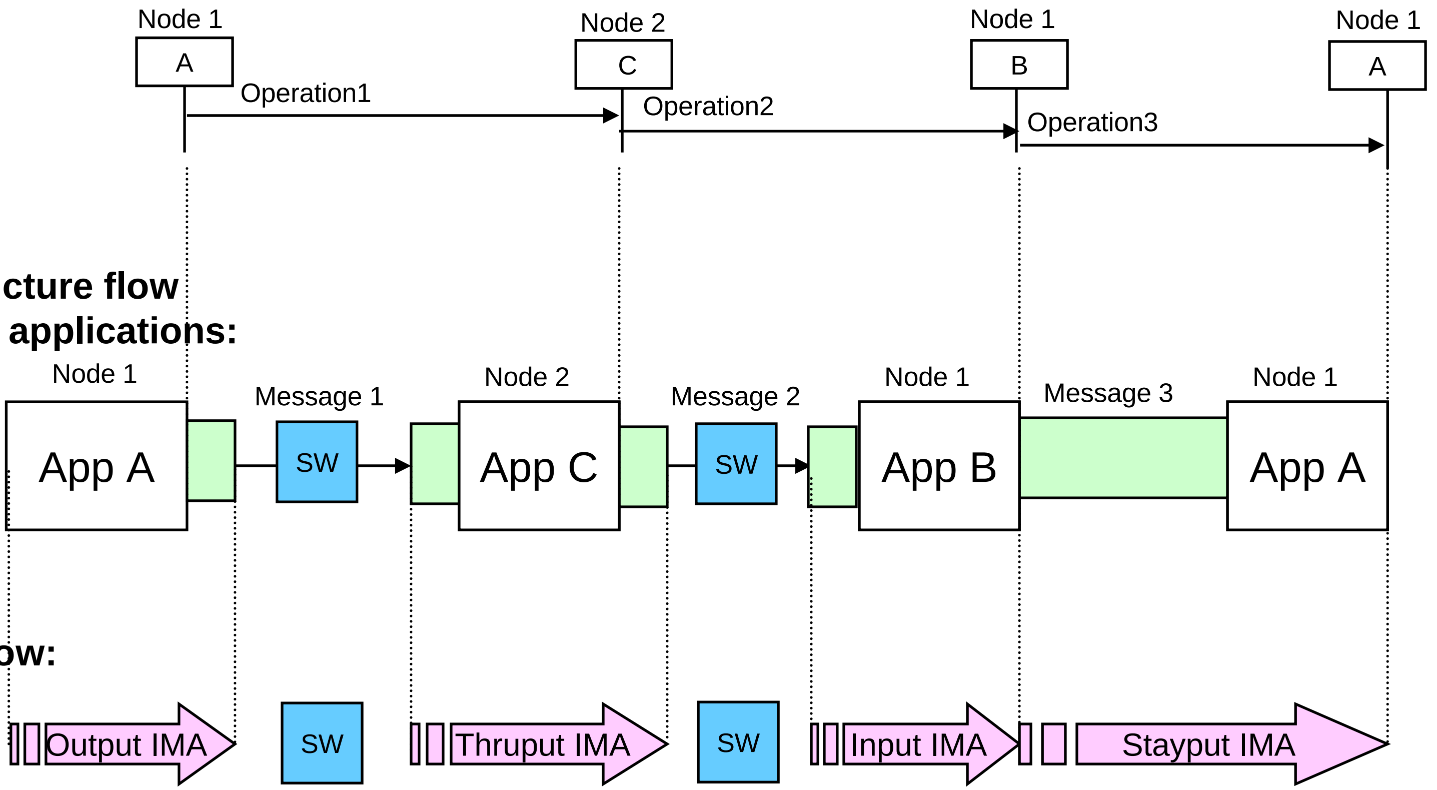
September 30, 2004

Example: UML Sequence Diagram with Added Architecture Detail



Sequence Diagram Flows can be Interpreted in Terms of Infrastructure Modeling Assemblies (IMAs)

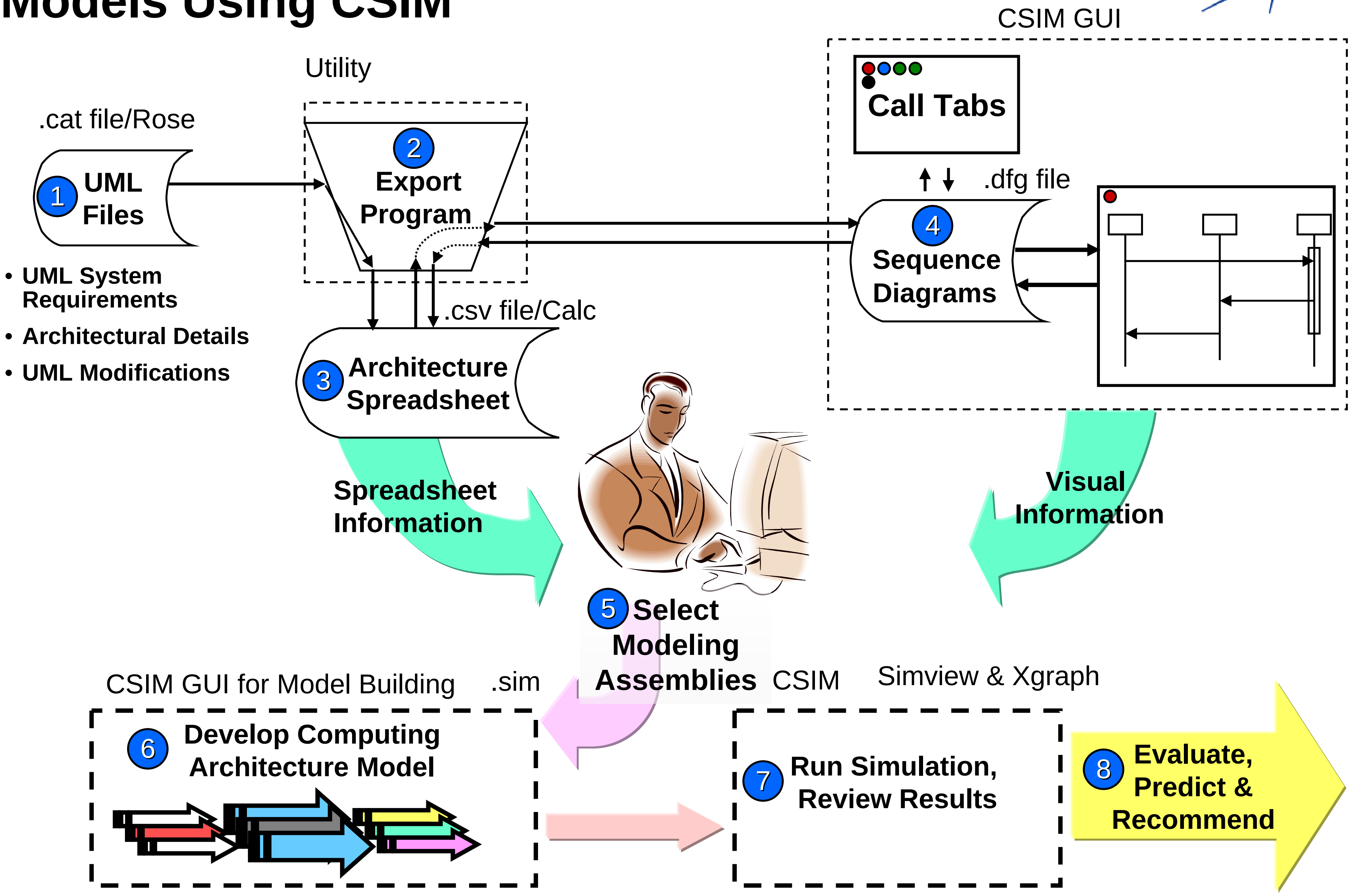
- **Operations on the sequence diagram:**
- **Infrastructure flow between applications:**
- **Model flow:**



SW ⇒ Network Switch

🟩 ⇒ Middleware &/or internet protocol

Building and Executing Performance Models Using CSIM



.cat file/Rose
1 UML Files
Utility
2 Export Program
.csv file/Calc
3 Architecture Spreadsheet
CSIM GUI
Call Tabs
.dfg file
4 Sequence Diagrams
Spreadsheet Information
Visual Information
5 Select Modeling Assemblies
CSIM GUI for Model Building
.sim
CSIM
Simview & Xgraph
6 Develop Computing Architecture Model
7 Run Simulation, Review Results
8 Evaluate, Predict & Recommend

- UML System Requirements
- Architectural Details
- UML Modifications

LOCKHEED MARTIN